Лабораторна робота №2

Тема: Виконання дій у системі за допомогою скриптів.

Написати пакетний файл який:

1. **Має аргументи командного рядка:** ***Аргумент1, Аргумент2, Аргумент3, Аргумент4, Аргумент5, Аргумент6.***

Лістинг:

```
2. ' Перевірка кількості переданих аргументів
3. If WScript.Arguments.Count <> 6 Then
4.     WScript.Echo "Неправильна кількість аргументів. Перевірте вхідні
   дані."
5.     WScript.Quit 1
6. End If
7.
8. ' Встановлення значень аргументів
9. Arg1 = WScript.Arguments(0)
10.  Arg2 = WScript.Arguments(1)
11.  Arg3 = WScript.Arguments(2)
12.  Arg4 = WScript.Arguments(3)
13.  Arg5 = WScript.Arguments(4)
14.  Arg6 = WScript.Arguments(5)
```

Перевіряє чи існує файл, ім’я якого завдано у *Аргумент1*.
Якщо не існує, то створює його. Це буде log файл скрипта.
Дописує у цей файл :

поточну дату та час;

«Файл з ім’ям *Аргумент1* відкрито або створено».

Лістинг:

```
' Отримання ім'я файлу з першого аргументу
fileName = WScript.Arguments(0)

' Створення об'єкту для роботи з файловою системою
Set fso = CreateObject("Scripting.FileSystemObject")

' Перевірка існування файлу
If Not fso.FileExists(fileName) Then
    ' Створення файлу, якщо він не існує
    Set logFile = fso.CreateTextFile(fileName, True)
    ' Запис поточної дати та часу у файл
    logFile.WriteLine "Дата і час: " & Now
    logFile.WriteLine "Файл з ім'ям " & fileName & " відкрито або створено."
    logFile.Close
    WScript.Echo "Файл '" & fileName & "' створено."
Else
    ' Дописування у файл, якщо він вже існує
    Set logFile = fso.OpenTextFile(fileName, 8, True)
    logFile.WriteLine "Дата і час: " & Now
    logFile.WriteLine "Файл з ім'ям " & fileName & " відкрито або створено."
    logFile.Close
    WScript.Echo "Файл '" & fileName & "' вже існує. Дані додано."
End If
```

Цей скрипт на мові VBScript отримує ім'я файлу з першого аргументу командного рядка, перевіряє, чи існує вказаний файл, і виконує наступні дії:

1. **Отримання імені файлу з першого аргументу**: Ім'я файлу зчитується з першого аргументу командного рядка.
2. **Створення об'єкту для роботи з файловою системою**: Використовується об'єкт `Scripting.FileSystemObject` для роботи з файлами та папками.
3. **Перевірка існування файлу**: Виконується перевірка існування файлу за вказаним ім'ям.
4. **Створення файлу, якщо він не існує**: Якщо файл не існує, створюється новий файл за вказаним ім'ям. У нього записується поточна дата та час, інформація про створення або відкриття файлу, після чого файл закривається.
5. **Дописування у файл, якщо він вже існує**: Якщо файл вже існує, до нього дописується поточна дата та час, інформація про відкриття або створення файлу, після чого файл закривається.
6. **Вивід повідомлення в консоль**: Залежно від того, чи був створений новий файл чи додані дані до існуючого, скрипт виводить відповідне повідомлення у консоль.

Отримати час з NTP серверу та встановити його поточним. Записати оновлений час у log.

Лістинг:

```
' Запуск зовнішньої команди для оновлення часу з NTP серверу
Set objShell = CreateObject("WScript.Shell")
objShell.Run "w32tm /resync"

' Отримання поточного часу
currentTime = Now

' Отримання ім'я файлу з другого аргументу
fileName = WScript.Arguments(1)

' Відкриття або створення файлу log
Set fso = CreateObject("Scripting.FileSystemObject")
Set logFile = fso.OpenTextFile(fileName, 8, True)

' Запис поточного часу у файл log
logFile.WriteLine "Поточний час: " & currentTime

' Закриття файлу log
logFile.Close
```

Цей скрипт на мові VBScript запускає зовнішню команду для оновлення часу з NTP серверу, потім отримує поточний час, записує його у файл логу та закриває файл логу.

1. **Запуск зовнішньої команди для оновлення часу з NTP серверу**: Використовується об'єкт `WScript.Shell` для виклику команди `w32tm /resync`, яка оновлює час з NTP серверу.
2. **Отримання поточного часу**: Використовується функція `Now` для отримання поточного часу.

3. **Отримання імені файлу з другого аргументу**: Зчитується другий аргумент командного рядка, який вказує на ім'я файлу логу.
4. **Відкриття або створення файлу log**: Створюється або відкривається файл логу для запису поточного часу.
5. **Запис поточного часу у файл log**: Поточний час записується у файл логу.
6. **Закриття файлу log**: Файл логу закривається після запису поточного часу у нього.

Виводить у log список усіх запущених процесів.

Лістинг:

```
' Отримання ім'я файлу з другого аргументу
fileName = WScript.Arguments(1)

' Відкриття або створення файлу log
Set fso = CreateObject("Scripting.FileSystemObject")
Set logFile = fso.OpenTextFile(fileName, 8, True)

' Вивід списку усіх запущених процесів у файл log
Set objShell = CreateObject("WScript.Shell")
Set objExec = objShell.Exec("tasklist")
Do Until objExec.StdOut.AtEndOfStream
    strLine = objExec.StdOut.ReadLine
    logFile.WriteLine strLine
Loop

' Закриття файлу log
logFile.Close
```

Цей скрипт на мові VBScript отримує ім'я файлу з другого аргументу командного рядка, відкриває або створює файл логу, записує усі запущені процеси у цей файл логу, а потім закриває файл логу.

1. **Отримання імені файлу з другого аргументу**: Код отримує другий аргумент з командного рядка, який представляє ім'я файлу логу.
2. **Відкриття або створення файлу log**: Створюється або відкривається файл логу для запису усієї інформації про запущені процеси.
3. **Вивід списку усіх запущених процесів у файл log**: За допомогою команди `tasklist` отримується список усіх запущених процесів, який потім записується у файл логу.
4. **Закриття файлу log**: Файл логу закривається після того, як весь список запущених процесів було записано у нього.

Завершує процес, з ім’ям *Аргумент3*. Інформує в log.

Лістинг:

```
' Отримання третього аргументу - імені процесу для завершення
processName = WScript.Arguments(2)

' Відкриття або створення файлу log
Set fso = CreateObject("Scripting.FileSystemObject")
Set logFile = fso.OpenTextFile(fileName, 8, True)
```

```
' Завершення процесу
Set objShell = CreateObject("WScript.Shell")
objShell.Run "taskkill /F /IM " & processName, 0, True

' Запис інформації про завершення процесу у файл log
logFile.WriteLine "Завершено процес з ім'ям " & processName

' Закриття файлу log
logFile.Close
```

Цей скрипт на мові VBScript завершує процес з вказаним іменем та записує інформацію про це у лог-файл.

1. **Отримання третього аргументу - імені процесу для завершення**: Код отримує третій аргумент з командного рядка, який представляє ім'я процесу, який потрібно завершити.
2. **Відкриття або створення файлу log**: Створюється або відкривається лог-файл для запису інформації про завершення процесу. Ім'я файлу логу було отримано як перший аргумент командного рядка.
3. **Завершення процесу**: Викликається команда `taskkill /F /IM`, яка завершує процес з вказаним іменем. Опція `/F` вказує на примусове завершення процесу.
4. **Запис інформації про завершення процесу у файл log**: Записується інформація про завершення процесу у лог-файл.
5. **Закриття файлу log**: Лог-файл закривається після запису інформації про завершення процесу.

Видаляє усі файли за шляхом *Аргумент2*, які мають розширення .TMP, або їх ім’я починається на «temp».

Інформацію про виконані дії записує у log файл. Вказати кількість видалених файлів.

Лістинг:

```
' Отримання другого аргументу - шляху до файлів для видалення
Dim folderPath
folderPath = WScript.Arguments(1)

' Створення об'єкту FileSystemObject
Dim fso
Set fso = CreateObject("Scripting.FileSystemObject")

' Лог файл
Dim logFilePath
logFilePath = WScript.Arguments(0)

' Відкриття лог файлу для дописування
Dim logFile
Set logFile = fso.OpenTextFile(logFilePath, 8, True)


' Логіка видалення файлів
Dim deletedCount
deletedCount = 0
deletedCount = deletedCount + DeleteFiles(folderPath, ".TMP")
deletedCount = deletedCount + DeleteTempFiles(folderPath, "temp")

' Запис кількості видалених файлів у лог файл
```

```
logFile.WriteLine "Кількість видалених файлів: " & deletedCount

' Закриття лог файлу
logFile.Close

' Підпрограма для видалення файлів з певним розширенням
Function DeleteFiles(folderPath, extension)
    Dim folder, file, files
    Set folder = fso.GetFolder(folderPath)
    Set files = folder.Files
    Dim count
    count = 0
    For Each file in files
        If LCase(fso.GetExtensionName(file.Name)) = LCase(extension) Then
            file.Delete True ' True - видалити файл без запиту підтвердження
            count = count + 1
        End If
    Next
    DeleteFiles = count
End Function

' Підпрограма для видалення файлів з певним початком імені
Function DeleteTempFiles(folderPath, startName)
    Dim folder, file, files
    Set folder = fso.GetFolder(folderPath)
    Set files = folder.Files
    Dim count
    count = 0
    For Each file in files
        If LCase(Left(file.Name, Len(startName))) = LCase(startName) Then
            file.Delete True ' True - видалити файл без запиту підтвердження
            count = count + 1
        End If
    Next
    DeleteTempFiles = count
End Function
```

Цей скрипт на мові VBScript видаляє файли з вказаної папки за певними умовами та записує кількість видалених файлів у лог-файл.

1. **Отримання другого аргументу - шляху до файлів для видалення**: Код отримує другий аргумент з командного рядка, який представляє шлях до папки з файлами, які потрібно видалити.
2. **Створення об'єкту FileSystemObject**: Створюється об'єкт `FileSystemObject`, який потрібен для роботи з файловою системою.
3. **Відкриття лог файлу для дописування**: Відкривається лог-файл для дописування. Шлях до лог-файлу був отриманий як перший аргумент командного рядка.
4. **Логіка видалення файлів**:
    - Викликається функція `DeleteFiles`, яка видаляє файли з певним розширенням (`.TMP`).
    - Викликається функція `DeleteTempFiles`, яка видаляє файли з певним початком імені (`temp`).
    - Кількість видалених файлів підраховується та записується у лог-файл.
5. **Закриття лог файлу**: Лог-файл закривається після запису інформації про видалені файли.
6. **Підпрограми для видалення файлів**:
    - `DeleteFiles`: Видаляє файли з певним розширенням (`.TMP`) у вказаній папці.
    - `DeleteTempFiles`: Видаляє файли з певним початком імені (`temp`) у вказаній папці.

Стискає усі файли які залишилися за шляхом *Аргумент2* у архів .zip. Ім’я архіву – поточна дата та час

Лістинг:

```
' Отримання третього аргументу - шляху до архіву
Dim archivePath
archivePath = WScript.Arguments(2)

' Створення об'єкту для роботи з Shell
Dim objShell
Set objShell = CreateObject("Shell.Application")

' Стискаємо файли
ZipFiles archivePath, WScript.Arguments(1)

' Підпрограма для стискання файлів
Sub ZipFiles(archivePath, folderPath)
    Dim sourceFolder, files, file
    Set sourceFolder = objShell.NameSpace(folderPath)
    Set files = sourceFolder.Items
    objShell.Namespace(archivePath & ".zip").CopyHere files
End Sub
```

Цей скрипт на мові VBScript стискає файли у вказаній папці та зберігає їх у вказаному архіві.

1. **Отримання третього аргументу - шляху до архіву**: Код отримує третій аргумент з командного рядка, який представляє шлях до архіву.
2. **Створення об'єкту для роботи з Shell**: Створюється об'єкт `Shell.Application`, який потрібен для роботи з архівами.
3. **Стискаємо файли**: Викликається підпрограма `ZipFiles`, якій передається шлях до архіву (третій аргумент командного рядка) та шлях до папки з файлами, які потрібно стиснути (другий аргумент командного рядка).
4. **Підпрограма для стискання файлів**: Ця підпрограма використовує об'єкт `Shell.Application` для стискання файлів. Вона приймає шлях до архіву (`archivePath`) та шлях до папки, в якій знаходяться файли, які потрібно стиснути (`folderPath`). Файли з папки копіюються до архіву, який знаходиться за вказаним шляхом, за допомогою методу `CopyHere`.

Переписує створений архів у папку за шляхом *Аргумент4*.

Лістинг:

```
' Отримання четвертого аргументу - шляху до папки для переписування архіву
Dim destinationFolder
destinationFolder = WScript.Arguments(3)

' Переписування архіву
CopyFile WScript.Arguments(2) & ".zip", destinationFolder

' Підпрограма для копіювання файлу
Sub CopyFile(sourceFile, destFolder)
    Dim objFSO
    Set objFSO = CreateObject("Scripting.FileSystemObject")
    objFSO.CopyFile sourceFile, destFolder & "\", True
End Sub
```

Цей код на мові VBScript отримує шлях до папки, у яку потрібно переписати архів, як четвертий аргумент командного рядка. Потім архів копіюється у вказану папку.

1. **Отримання четвертого аргументу - шляху до папки для переписування архіву**: Код отримує четвертий аргумент з командного рядка, який представляє собою шлях до папки, у яку потрібно переписати архів.
2. **Переписування архіву**: Викликається підпрограма `CopyFile`, якій передається шлях до архіву (третій аргумент командного рядка з розширенням ".zip") та шлях до папки призначення (записаний у змінну `destinationFolder`).
3. **Підпрограма для копіювання файлу**: Ця підпрограма використовує об'єкт `FileSystemObject` для копіювання файлу. Вона приймає шлях до джерела (аргумент `sourceFile`) та шлях до папки призначення (аргумент `destFolder`). Параметр `True` вказує, що якщо файл існує у папці призначення, він буде замінений.

Перевіряє чи є файл з архівом за минулий день. Інформує у log.Якщо нема, інформує на email.

Лістинг:

```
' Перевірка чи існує архів за минулий день
Dim yesterdayArchivePath
yesterdayArchivePath = destinationFolder & "\" & FormatDateTime(Date - 1, 2) &
".zip"

Dim objFSO
Set objFSO = CreateObject("Scripting.FileSystemObject")

If objFSO.FileExists(yesterdayArchivePath) Then
    WriteToLog "Знайдено архів за минулий день: " & yesterdayArchivePath
Else
    WriteToLog "Архів за минулий день не знайдено"
    SendEmail "user@example.com", "Відсутній архів за минулий день", "Архів за
минулий день відсутній."
End If

' Підпрограма для запису в лог
Sub WriteToLog(logMessage)
    Dim logFile, fso, ts
    Set fso = CreateObject("Scripting.FileSystemObject")
    Set logFile = fso.OpenTextFile(WScript.Arguments(1), 8, True)
    logFile.WriteLine Now & " - " & logMessage
    logFile.Close
End Sub

' Підпрограма для відправки електронного листа
Sub SendEmail(emailAddress, subject, body)
    Dim objEmail
    Set objEmail = CreateObject("CDO.Message")

    ' Налаштування відправника та отримувача
    objEmail.From = "vlad.negerey1@google.com"
    objEmail.To = emailAddress
    objEmail.Subject = subject
    objEmail.TextBody = body

    ' Налаштування SMTP сервера
    objEmail.Configuration.Fields.Item
("http://schemas.microsoft.com/cdo/configuration/sendusing") = 2
```

```
    objEmail.Configuration.Fields.Item
("http://schemas.microsoft.com/cdo/configuration/smtpserver") =
"smtp.example.com"
    objEmail.Configuration.Fields.Item
("http://schemas.microsoft.com/cdo/configuration/smtpserverport") = 25
    objEmail.Configuration.Fields.Update

    ' Відправлення листа
    objEmail.Send
End Sub
```

Цей код на мові VBScript перевіряє наявність архіву за минулий день у вказаній папці та повідомляє про це у лог-файлі. Якщо архів відсутній, відправляється електронний лист на вказану адресу.

1. **Перевірка чи існує архів за минулий день**: Код спочатку формує шлях до архіву за попередній день за допомогою `FormatDateTime(Date - 1, 2)`. Потім перевіряється наявність цього архіву за допомогою `objFSO.FileExists`. Якщо архів існує, відображається повідомлення у лог-файлі. Якщо архів відсутній, викликається підпрограма `SendEmail` для відправки електронного листа.
2. **Підпрограма для запису в лог**: Ця підпрограма відкриває лог-файл для запису та додає в нього поточну дату та час разом з переданим повідомленням.
3. **Підпрограма для відправки електронного листа**: Ця підпрограма використовує об'єкт `CDO.Message` для створення та відправлення електронного листа. Налаштовуються параметри листа (відправник, отримувач, тема, текст) та SMTP-сервера для відправлення листа.

Перевіряє, чи є за шляхом *Аргумент4*, архіви, старші 30 днів, та якщо є, то видаляє. Інформує в log.

Лістинг:

```
' Перевіряє, чи є за шляхом Аргумент4 архіви, старші 30 днів, та якщо є, то
видаляє. Інформує в log.
Sub DeleteOldArchives(folderPath)
    ' Створення об'єкта для роботи з файловою системою
    Dim objFSO : Set objFSO = CreateObject("Scripting.FileSystemObject")
    ' Перелік усіх файлів у папці
    Dim objFolder : Set objFolder = objFSO.GetFolder(folderPath)
    Dim objFiles : Set objFiles = objFolder.Files
    ' Поточна дата
    Dim currentDate : currentDate = Now
    ' Кількість днів, які вважаються застарілими
    Const DAYS_THRESHOLD = 30

    ' Перегляд усіх файлів у папці
    For Each objFile In objFiles
        ' Перевірка, чи файл є архівом .zip
        If LCase(objFSO.GetExtensionName(objFile.Name)) = "zip" Then
            ' Обчислення різниці у датах (кількість днів)
            Dim daysDifference : daysDifference = DateDiff("d",
objFile.DateLastModified, currentDate)
            ' Якщо файл старший за визначений поріг, видаляємо його
            If daysDifference > DAYS_THRESHOLD Then
                objFSO.DeleteFile objFile.Path, True ' Видалення файлу
                WriteToLog "Видалено застарілий архів: " & objFile.Name
            End If
        End If
    Next
End Sub
```

Цей код на мові VBScript перевіряє наявність старших за 30 днів архівів у вказаній папці та видаляє їх, інформуючи про це у лог-файлі.

1. **Sub DeleteOldArchives(folderPath)**: Ця підпрограма переглядає всі файли у вказаній папці. Для кожного файлу, який має розширення .zip, обчислюється різниця у датах між поточною датою та датою останньої зміни файлу. Якщо ця різниця перевищує 30 днів (заданий поріг), то файл видаляється за допомогою методу `DeleteFile` об'єкта `FileSystemObject`. Після цього в лог-файл записується повідомлення про видалення старого архіву.

Перевіряє чи є підключення до Internet, та інформує в log.

Для перевірки підключення до Інтернету можна використати простий запит до веб-ресурсу, наприклад, сайту Google, і перевірити, чи відбулася успішна відповідь.

Лістинг:

```
' Перевіряє, чи є підключення до Internet, та інформує в log.
Sub CheckInternetConnection()
    ' Створення об'єкта для виконання HTTP-запитів
    Dim objHTTP : Set objHTTP = CreateObject("MSXML2.ServerXMLHTTP")
    ' URL-адреса для перевірки доступності Інтернету
    Dim url : url = "http://www.google.com"

    ' Виконання HTTP-запиту за вказаною URL-адресою
    objHTTP.Open "GET", url, False
    objHTTP.send

    ' Перевірка статусу HTTP-відповіді
    If objHTTP.Status = 200 Then
        WriteToLog "Підключення до Інтернету є."
    Else
        WriteToLog "Немає підключення до Інтернету."
    End If
End Sub
```

Цей код на мові VBScript перевіряє наявність підключення до Інтернету та інформує про це у лог-файлі.

1. **Sub CheckInternetConnection()**: Ця підпрограма виконує перевірку доступності Інтернету. Вона створює об'єкт для виконання HTTP-запитів за допомогою `MSXML2.ServerXMLHTTP`, після чого відкриває з'єднання за вказаною URL-адресою (зазвичай це `http://www.google.com`). Після відправлення запиту перевіряється статус HTTP-відповіді. Якщо статус рівний 200, це означає успішне з'єднання, тому в лог-файл записується повідомлення "Підключення до Інтернету є.". У протилежному випадку записується повідомлення "Немає підключення до Інтернету.".

Перевірити чи є в локальній мережі комп’ютер з IP *аргумент5*, та якщо є, то завершити його роботу. Інформувати в log.

Для перевірки наявності комп'ютера з вказаною IP-адресою у локальній мережі та завершення його роботи можна використати команду ping. Якщо

комп'ютер з вказаною IP-адресою доступний у мережі, то відповідь на пінг буде успішною.

Лістинг:

```
' Перевіряє наявність комп'ютера з вказаною IP-адресою у локальній мережі та завершує його роботу.
Sub CheckAndShutdownComputer(ipAddress)
    ' Виконання команди ping для перевірки доступності комп'ютера з вказаною IP-адресою
    Dim command : command = "ping -n 1 " & ipAddress
    Dim shell : Set shell = CreateObject("WScript.Shell")
    Dim pingResult : pingResult = shell.Run(command, 0, True)

    ' Перевірка результату виконання команди ping
    If pingResult = 0 Then
        WriteToLog "Комп'ютер з IP-адресою " & ipAddress & " знайдено у мережі. Завершення роботи..."
        ' Виконання команди для завершення роботи комп'ютера
        shell.Run "shutdown -s -f -t 0", 0, True
    Else
        WriteToLog "Комп'ютер з IP-адресою " & ipAddress & " не знайдено у мережі."
    End If
End Sub
```

Цей код на мові VBScript виконує такі дії:

1. **Sub CheckAndShutdownComputer(ipAddress)**: Ця підпрограма перевіряє наявність комп'ютера з вказаною IP-адресою у локальній мережі та завершує його роботу, якщо комп'ютер доступний. Вона використовує команду `ping -n 1 [IP-адреса]` для перевірки доступності комп'ютера. Якщо команда `ping` успішно виконується (результатом є 0), це означає, що комп'ютер доступний у мережі, і виконується команда `shutdown -s -f -t 0` для завершення його роботи. Результати перевірки записуються у лог-файл за допомогою підпрограми `WriteToLog`.
2. **WriteToLog(message)**: Ця підпрограма записує передане повідомлення у лог-файл. Вона відкриває файл логу, додає до нього поточну дату та час, а також передане повідомлення, та закриває файл.

Отримує список комп’ютерів в мережі та записує отриману інформацію у log.

Для отримання списку комп'ютерів у мережі можна скористатися командою arp -a у командному рядку. Ця команда показує таблицю ARP (Address Resolution Protocol), яка містить інформацію про зв'язки між IP-адресами та MAC-адресами пристроїв у мережі.

Лістинг:

```
' Отримує список комп'ютерів у мережі та записує отриману інформацію у log.
Sub GetNetworkComputers()
    ' Виконання команди arp -a для отримання списку комп'ютерів у мережі
    Dim shell : Set shell = CreateObject("WScript.Shell")
    Dim command : command = "arp -a"
    Dim result : result = shell.Exec(command)

    ' Зчитування результату виконання команди та запис у log
    While Not result.StdOut.AtEndOfStream
```

```
        Dim line : line = result.StdOut.ReadLine
        WriteToLog line
    Wend

    WriteToLog "Список комп'ютерів у мережі успішно отримано."
End Sub
```

Цей скрипт на мові VBScript виконує наступні дії:

1. **Sub GetNetworkComputers()**: Ця підпрограма виконує команду `arp -a` для отримання списку комп'ютерів у мережі. Вона використовує об'єкт `WScript.Shell` для запуску командного рядка. Результати виконання команди зчитуються рядок за рядком і записуються у лог-файл за допомогою підпрограми `WriteToLog`.
2. **WriteToLog(message)**: Ця підпрограма записує передане повідомлення у лог-файл. Вона відкриває файл логу, додає до нього поточну дату та час, а також передане повідомлення, та закриває файл.

Перевіряє, якщо один з комп’ютерів, зі вказаними IP адресами у файлі ipon.txt не присутній в мережі, інформує в log та на email.

Для перевірки доступності комп'ютерів у мережі за вказаними IP-адресами ми можемо використати команду ping у VBScript. Вона дозволяє визначити, чи доступний комп'ютер за його IP-адресою.

Лістинг:

```
' Функція для перевірки доступності комп'ютера за IP адресою
Function CheckComputerAvailability(ipAddress)
    Dim objShell : Set objShell = CreateObject("WScript.Shell")
    Dim objPing : Set objPing =
GetObject("winmgmts:{impersonationLevel=impersonate}").ExecQuery("select * from
Win32_PingStatus where address = '" & ipAddress & "'")

    For Each objResult In objPing
        If IsObject(objResult) Then
            If objResult.StatusCode = 0 Then
                ' Комп'ютер доступний у мережі
                CheckComputerAvailability = True
            Else
                ' Комп'ютер не доступний у мережі
                CheckComputerAvailability = False
            End If
        End If
    Next
End Function

' Функція для запису повідомлення у лог
Sub WriteToLog(message)
    Dim fso : Set fso = CreateObject("Scripting.FileSystemObject")
    Dim logFile : Set logFile = fso.OpenTextFile("lab.log", 8, True)
    logFile.WriteLine Now & " " & message
    logFile.Close
End Sub

' Читання IP адресів з файлу
Dim ipFile : Set ipFile =
CreateObject("Scripting.FileSystemObject").OpenTextFile("ipon.txt", 1)
Dim ipAddress : ipAddress = ipFile.ReadAll
ipFile.Close
```

```
' Перетворення рядка в масив IP адрес
Dim ipAddresses : ipAddresses = Split(ipAddress, vbCrLf)

' Перевірка доступності кожного IP адресу
Dim availableComputers : availableComputers = ""
Dim unavailableComputers : unavailableComputers = ""
For Each ip In ipAddresses
    If CheckComputerAvailability(ip) Then
        availableComputers = availableComputers & ip & vbCrLf
    Else
        unavailableComputers = unavailableComputers & ip & vbCrLf
    End If
Next

' Запис до лог файлу
WriteToLog "Доступні комп'ютери:"
WriteToLog availableComputers
WriteToLog "Недоступні комп'ютери:"
WriteToLog unavailableComputers
```

Цей скрипт на мові VBScript виконує наступні дії:

1. **Function CheckComputerAvailability(ipAddress)**: Ця функція перевіряє доступність комп'ютера за його IP-адресою. Вона використовує об'єкт `Win32_PingStatus` для виконання ICMP запиту до заданої IP-адреси. Якщо відповідь на запит успішна (код статусу 0), то функція повертає значення `True`, інакше - `False`.
2. **Sub WriteToLog(message)**: Ця підпрограма призначена для запису повідомлення у лог-файл. Вона відкриває файл логу, додає до нього поточну дату та час, а також передане повідомлення, та закриває файл.
3. **Читання IP адресів з файлу**: Зчитує IP адреси з файлу "ipon.txt".
4. **Перетворення рядка в масив IP адрес**: Розділяє рядок з IP адресами на окремі адреси.
5. **Перевірка доступності кожного IP адресу**: Для кожної IP адреси з масиву викликається функція `CheckComputerAvailability`, яка перевіряє її доступність у мережі. Результати перевірки записуються у відповідні змінні.
6. **Запис до лог файлу**: Записує результати перевірки доступності комп'ютерів у лог-файл.

Перевіряє, якщо розмір поточного log файлу більший за *Аргумент6*, то інформує в log та на email.

Щоб реалізувати цю функціональність, спочатку нам потрібно отримати розмір поточного лог-файлу. Після цього порівняємо його з заданим лімітом, який передається в аргументі.

Лістинг:

```
' Функція для перевірки розміру лог файлу
Function CheckLogFileSize(logFilePath, maxSize)
    Dim fso : Set fso = CreateObject("Scripting.FileSystemObject")
    Dim logFile, fileSize

    ' Перевірка наявності файлу
    If fso.FileExists(logFilePath) Then
        ' Отримання об'єкта файлу
        Set logFile = fso.GetFile(logFilePath)
        ' Отримання розміру файлу
        fileSize = logFile.Size
```

```
        ' Перевірка розміру файлу
        If fileSize > maxSize Then
            ' Інформування в лог
            WriteToLog "Розмір лог-файлу перевищує допустимий ліміт (" & maxSize
& " байт)."
        End If
    Else
        ' Інформування в лог про відсутність файлу
        WriteToLog "Файл логу не знайдено: " & logFilePath
    End If

    ' Звільнення ресурсів
    Set fso = Nothing
End Function
```

Ця функція написана на мові VBScript і призначена для перевірки розміру лог-файлу. Вона виконує наступні дії:

1. **Function CheckLogFileSize(logFilePath, maxSize)**: Ця функція перевіряє розмір заданого лог-файлу та порівнює його з максимальним допустимим розміром.
2. **Dim fso, logFile, fileSize**: В цьому рядку оголошуються змінні, які використовуються для взаємодії з файловою системою.
3. **If fso.FileExists(logFilePath) Then**: Перевірка існування лог-файлу за заданим шляхом.
4. **Set logFile = fso.GetFile(logFilePath)**: Отримання об'єкта файлу за заданим шляхом.
5. **fileSize = logFile.Size**: Отримання розміру лог-файлу.
6. **If fileSize > maxSize Then**: Перевірка, чи перевищує розмір лог-файлу максимально допустимий розмір.
7. **WriteToLog "Розмір лог-файлу перевищує допустимий ліміт (" & maxSize & " байт)."**: Якщо розмір перевищує допустимий ліміт, то ця функція записує в лог-файл відповідне повідомлення.
8. **WriteToLog "Файл логу не знайдено: " & logFilePath**: Якщо лог-файл не знайдено, то ця функція записує в лог-файл відповідне повідомлення.
9. **Set fso = Nothing**: Звільнення ресурсів.

Перевіряє кількість вільного та зайнятого місця на усіх дисках в системі та пише цю інформацію в log.

Для перевірки кількості вільного та зайнятого місця на усіх дисках можна використати WMI (Windows Management Instrumentation)

Лістинг:

```
' Функція для отримання інформації про диски
Function GetDiskSpaceInfo()
    Dim objWMIService, colDisks, objDisk, info
    Dim fso : Set fso = CreateObject("Scripting.FileSystemObject")
    Dim logMessage

    ' Підключення до WMI
    Set objWMIService = GetObject("winmgmts:\\.\root\CIMV2")
    Set colDisks = objWMIService.ExecQuery("SELECT * FROM Win32_LogicalDisk")

    ' Перебір дисків
    For Each objDisk In colDisks
        ' Отримання інформації про диск
```

```
        info = "Диск " & objDisk.DeviceID & ": " & FormatSize(objDisk.FreeSpace)
& " вільно з " & FormatSize(objDisk.Size)
        ' Запис інформації про диск у лог
        WriteToLog info
    Next

    ' Звільнення ресурсів
    Set objWMIService = Nothing
    Set colDisks = Nothing
    Set fso = Nothing
End Function

' Функція для форматування розміру у зручний для читання вигляд
Function FormatSize(size)
    Dim units : units = Array("Б", "КБ", "МБ", "ГБ", "ТБ")
    Dim i : i = 0

    Do While size >= 1024 And i < UBound(units)
        size = size / 1024
        i = i + 1
    Loop

    FormatSize = Round(size, 2) & " " & units(i)
End Function
```

Цей код написаний на мові VBScript і виконує наступні дії:

1. **Функція GetDiskSpaceInfo()**: Ця функція отримує інформацію про диски на комп'ютері та записує цю інформацію до лог-файлу. Вона використовує WMI (Windows Management Instrumentation) для отримання інформації про диски із системи.
2. **Dim objWMIService, colDisks, objDisk, info, fso, logMessage**: Оголошуються змінні для роботи з WMI, колекцією дисків, кожним диском, інформацією про диск, об'єктом FileSystemObject та повідомленням для лог-файлу.
3. **Set objWMIService = GetObject("winmgmts:\.\root\CIMV2")**: Підключення до WMI і отримання об'єкту `winmgmts`, який дозволяє взаємодіяти з сервісами управління Windows.
4. **Set colDisks = objWMIService.ExecQuery("SELECT * FROM Win32_LogicalDisk")**: Виконання запиту WMI для отримання інформації про всі логічні диски на комп'ютері.
5. **For Each objDisk In colDisks ... Next**: Цикл, який перебирає кожен об'єкт диска у колекції `colDisks`.
6. **info = "Диск " & objDisk.DeviceID & ": " & FormatSize(objDisk.FreeSpace) & " вільно з " & FormatSize(objDisk.Size)**: Формування рядка `info` з інформацією про диск, таку як його ідентифікатор, вільне та загальне місце.
7. **WriteToLog info**: Виклик підпрограми `WriteToLog`, яка записує інформацію про диск у лог-файл.
8. **Function FormatSize(size)**: Ця функція форматує розмір у зручний для читання вигляд (наприклад, "100 МБ"). Вона приймає розмір у байтах як вхідний параметр і повертає рядок зі зручним форматом.
9. **Set objWMIService = Nothing**, **Set colDisks = Nothing**, **Set fso = Nothing**: Звільнення ресурсів, щоб уникнути витоку пам'яті.

Записати результат виконання команди systeminfo у файл “*systeminfo+поточна дата-час.txt*”.

Лістинг:

```
' Функція для виконання команди systeminfo та запису результату у файл
Sub ExecuteSystemInfo()
    Dim objShell, objFSO, objFile, strCommand, strOutputFile

    ' Створення об'єктів Shell та FileSystemObject
    Set objShell = CreateObject("WScript.Shell")
    Set objFSO = CreateObject("Scripting.FileSystemObject")

    ' Формування команди systeminfo та шляху до вихідного файлу
    strCommand = "cmd /c systeminfo"
    strOutputFile = "systeminfo_" & FormatDateTime(Now, 1) & ".txt"

    ' Виконання команди systeminfo та запис результату у файл
    objShell.Run strCommand & " > " & strOutputFile, 0, True

    ' Перевірка, чи було створено файл
    If objFSO.FileExists(strOutputFile) Then
        WScript.Echo "Результати команди systeminfo записано у файл: " &
strOutputFile
    Else
        WScript.Echo "Не вдалося записати результати команди systeminfo у файл."
    End If

    ' Звільнення ресурсів
    Set objShell = Nothing
    Set objFSO = Nothing
End Sub

' Виклик функції для виконання команди systeminfo та запису результату у файл
ExecuteSystemInfo()
```

Цей скрипт на мові VBScript виконує команду systeminfo у командному рядку Windows і записує її результат у текстовий файл. Ось опис кожної частини коду:

1. **Sub ExecuteSystemInfo()**: Це оголошення підпрограми (підпрограма в VBScript називається Sub), яка виконує команду `systeminfo` і записує результат у файл.
2. **Dim objShell, objFSO, objFile, strCommand, strOutputFile**: Оголошення змінних, які будуть використовуватися для роботи з об'єктами Shell, FileSystemObject та для зберігання команди та імені вихідного файлу.
3. **Set objShell = CreateObject("WScript.Shell")** та **Set objFSO = CreateObject("Scripting.FileSystemObject")**: Ці рядки створюють об'єкти `WScript.Shell` і `Scripting.FileSystemObject`, які потрібні для виконання команди та запису результату у файл відповідно.
4. **strCommand = "cmd /c systeminfo"** та **strOutputFile = "systeminfo_" & FormatDateTime(Now, 1) & ".txt"**: Визначає команду `systeminfo`, яку слід виконати, та ім'я вихідного файлу, в який буде записаний результат команди. У цьому випадку, ім'я файлу буде містити поточну дату.
5. **objShell.Run strCommand & " > " & strOutputFile, 0, True**: Цей рядок виконує команду `systeminfo` у командному рядку (`cmd`) і перенаправляє її вивід у файл `strOutputFile`.
6. **If objFSO.FileExists(strOutputFile) Then ...**: Перевіряє, чи був створений файл з результатами команди `systeminfo`.
7. **WScript.Echo ...**: Виводить повідомлення про успішне або неуспішне завершення операції запису результатів команди у файл.
8. **Set objShell = Nothing** та **Set objFSO = Nothing**: Звільнює ресурси, щоб уникнути витоку пам'яті.
9. **ExecuteSystemInfo()**: Викликає підпрограму `ExecuteSystemInfo()` для виконання команди `systeminfo` та запису результату у файл.

Встановіть запуск вашого скрипту, регулярно, періодично.

```
Dim objShell
Set objShell = CreateObject("WScript.Shell")

' Шлях до VBScript файлу
scriptPath = "C:\Prog\laba\Script.vbs"

' Команда для створення завдання у Планувальнику завдань
taskCommand = "schtasks /create /sc daily /tn MyTask /tr """ & scriptPath & """
/st 09:00"

' Виконання команди
objShell.Run taskCommand, 0, True
```

Ця VBScript використовується для створення завдання в Планувальнику завдань Windows, яке запускатиме інший скрипт VBScript щоденно о 9:00 ранку. Ось опис кожної використаної команди:

1. **Dim objShell**: Ця команда створює змінну `objShell`, яка представляє об'єкт `WScript.Shell`.
2. **Set objShell = CreateObject("WScript.Shell")**: Ця команда створює екземпляр об'єкта `WScript.Shell`, який надає доступ до об'єкту командного рядка Windows.
3. **scriptPath = "C:\Prog\laba\Script.vbs"**: Це просто змінна, яка містить шлях до вашого VBScript файлу.
4. **taskCommand = "schtasks /create /sc daily /tn MyTask /tr """ & scriptPath & """ /st 09:00"**: Ця команда формує рядок команди для створення завдання в Планувальнику завдань. Параметри команди включають:
   - `/create`: Команда для створення нового завдання.
   - `/sc daily`: Частота запуску завдання (щоденно).
   - `/tn MyTask`: Ім'я створюваного завдання (MyTask).
   - `/tr "scriptPath"`: Шлях до скрипта, який буде виконувати завдання.
   - `/st 09:00`: Час запуску завдання (09:00).
5. **objShell.Run taskCommand, 0, True**: Ця команда запускає створення завдання у Планувальнику завдань за допомогою команди, яку ми побудували раніше. Параметри `0` та `True` вказують, що ми не хочемо відображати вікно командного рядка і чекати на завершення виконання команди.